3192 | L262 | 1

# calculator program

Input ~~Domain~~ Alphabet: { ~~Set of all~~ 0-9, ., (, ), +, -, #/, *, ^ }

Operations: { Addition, Subtraction, Division, multiplication, Exponentiation, Square Root, Convert Decimal to Fraction }

~~[1.] Square root and conversion into a fractional form shall be functions that are requested through an interface.~~

[2.] command line interface

[3.] How shall basic input be entered? (Basic meaning not requiring an explicit operation request). All but square root and fractional conversion can be requested implicitly. Square root requires either a defined input word or an interface request; the request requires a system of creating and tracking variables within the calculator, so expanding the input language shall suffice. Fractional conversion can only be performed on a value and not an operation, so it will be an interface request (though a complex implementation would define an input word that operates on only a single value and that cannot appear in an expression with other —— ... wait, this is simple).

In conclusion, input shall be entered within the program, and all operations shall be defined in the input language.

[4.] Keep Record the value of the last expression evaluated. (This requires a single variable in the calculator program.)

[5.] main $\xrightarrow{\text{input}}$ calc or $\xrightarrow{\text{input}}$ main $\xrightarrow{\text{values and order of operations}}$ calc

Either the calculator will parse the input string or the front-end will parse it and produce a collection of simple (two-value or one-value) expressions in the order they must be evaluated.

[6.] An expression can be decomposed into terms that represent simple two-value expressions (~~one~~ & and a term can contain simpler terms if needed).

~~Not Enough Time: allow only binary or unary operation expressions.~~

Perhaps there is enough time

---

[1.] Calculator depends on infix-to-prefix algorithm; ~~so~~ that ~~calculator~~ algorithm belongs ~~nowhere~~ nowhere but inside the calculator which depends on it for expression evaluation.

[2.] ~~Language~~ Alphabet

operations

struct Calculator
- infix-to-prefix algo
- most recent evaluation
- prefix evaluation algo
- input validation

3142 | Lab 2 | 2

[3.] Irrational conversion shall be an interface request and not defined in the input language.

[4.] Roots ~~will have~~ will have to be expressed as exponents.

---

After some time

[1.] main ← input

└ calculator(input)

calc.h
define struct

prefix
define prefix
algos